# PATENT ABSTRACTS OF JAPAN

(11)Publication number : **07-271764**
(43)Date of publication of application : **20.10.1995**

(51)Int.Cl. G06F 17/16

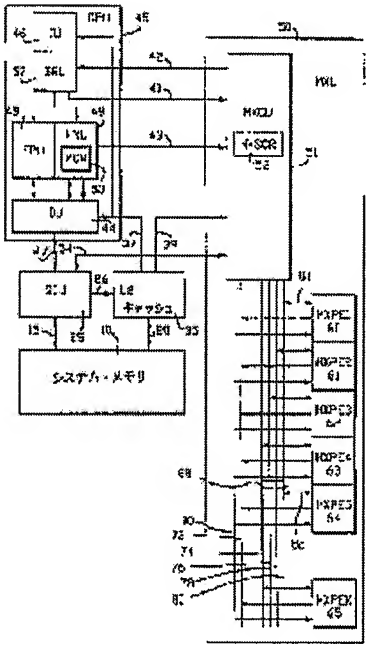
(21)Application number : **06-316800**
(22)Date of filing : **20.12.1994**

(71)Applicant : **INTERNATL BUSINESS MACH CORP <IBM>**
(72)Inventor : **ARGARWAL RAMESH CHANDRA**
**GROVES RANDALL DEAN**
**GUSTAVSON FRED GEHRUNG**
**JOHNSON MARK ALAN**
**OLSSON BRETT**

(30)Priority
Priority number : **94 217533** ????Priority date : **24.03.1994** ????Priority country : **US**

## (54) COMPUTER PROCESSOR AND SYSTEM

(57)Abstract:
PURPOSE: To provide a computer system provided with parallel processors for simultaneously executing an operation related to the group of data word pairs so as to substantially increase the throughput.
CONSTITUTION: A matrix processor for making high-speed numerical value calculation possible is presented. The processor is a vector processor formed from plural processing elements. A first processor is provided with a set of N pieces of registers and a first element or word among the N pieces of data vectors is stored in the registers. The respective processing elements are provided with an arithmetic unit capable of executing an arithmetic operation relating to the N pieces of elements in a set of N pieces of the registers. The respective data vectors are provided with the K pieces of the elements. Thus, the K pieces of the processing elements are present. For the vector operation of the matrix processor, the same operation is simultaneously executed relating to all the two or more vectors. Within one machine cycle after a preceding vector operation, a succeeding vector operation can be executed.



LEGAL STATUS

[Date of request for examination] 20.12.1994
[Date of sending the examiner s decision of rejection] 02.06.1998
[Kind of final disposal of application other than the examiner s decision of rejection or application converted registration]
[Date of final disposal for application]
[Patent number]
[Date of registration]
[Number of appeal against examiner s decision of 10-013597

rejection]

[Date of requesting appeal against examiner s decision of rejection] 28.08.1998

[Date of extinction of right]

(19)日本国特許庁（ＪＰ）　(12) 公開特許公報（A）　(11)特許出願公開番号

特開平7－271764

(43)公開日　平成７年(1995)10月20日

| (51)Int.Cl.[6] | 識別記号 | 庁内整理番号 | ＦＩ | 技術表示箇所 |
|---|---|---|---|---|
| Ｇ０６Ｆ　17/16 | | | | |
| | | | Ｇ０６Ｆ　15/ 347 | Ｓ |

審査請求　有　　請求項の数14　ＯＬ　（全 17 頁）

(21)出願番号　特願平6－316800

(22)出願日　平成６年(1994)12月20日

(31)優先権主張番号　２１７５３３

(32)優先日　1994年３月24日

(33)優先権主張国　米国（ＵＳ）

(71)出願人　390009531
インターナショナル・ビジネス・マシーンズ・コーポレイション
ＩＮＴＥＲＮＡＴＩＯＮＡＬ　ＢＵＳＩＮＥＳＳ　ＭＡＳＣＨＩＮＥＳ　ＣＯＲＰＯＲＡＴＩＯＮ
アメリカ合衆国10504、ニューヨーク州　アーモンク　（番地なし）

(72)発明者　ラメシュ・チャンドラ・アーガーワル
アメリカ合衆国ニューヨーク州、ヨークタウン・ハイツ、ガース・コート　650

(74)代理人　弁理士　合田　潔　（外２名）

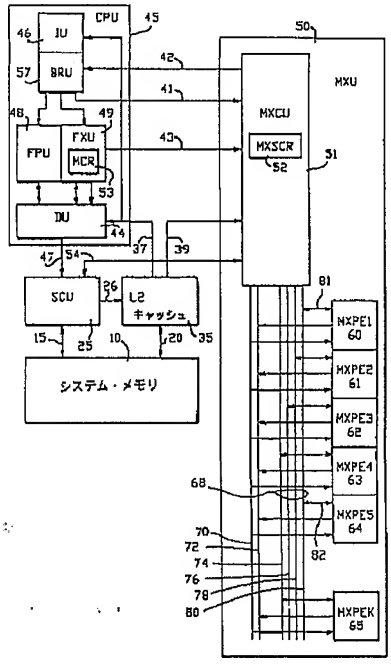
最終頁に続く

(54)【発明の名称】　計算機プロセッサ及びシステム

(57)【要約】

【目的】本発明は改良された計算機システムを提供する。

【構成】高速度数値計算を可能にするマトリクス処理装置が開示される。その処理装置は複数個の処理エレメントから形成されたベクトル処理装置である。Ｉ番目の処理装置はＮ個のレジスタのセットを有し、それらレジスタには、Ｎ個のデータ・ベクトルのうちのＩ番目のエレメント又はワードが記憶される。各処理エレメントは、Ｎ個のレジスタのセットにおけるＮ個のエレメントに関する演算オペレーションを遂行できる演算装置を有する。各データ・ベクトルはＫ個のエレメントを有する。従って、Ｋ個の処理エレメントが存在する。マトリクス処理装置のベクトル・オペレーションは２つ以上のベクトルすべてに関して同じオペレーションを同時に遂行する。先行のベクトル・オペレーションの後の１機械サイクル内で後続のベクトル・オペレーションが遂行可能である。



1

【特許請求の範囲】

【請求項1】プロセッサ・サイクル・タイムを有する計算機プロセッサにして、

K個の処理エレメント（但し、K＞1）にして、前記K個の処理エレメントの各々はN個のレジスタのセット（但し、N＞1）及び演算装置を含み、前記レジスタの各々のI番目のエレメントはベクトル長KのI番目のベクトルのための記憶ロケーションを提供し（但し、1≦I≦N）、それによって、N個のベクトルを提供するものと、

前記N個のベクトルの少なくとも1つの間でベクトル・オペレーションを所定のサイクル・タイムで遂行し、前記ベクトル・オペレーションの結果を前記N個のベクトルの1つに記憶するための手段にして、前記ベクトル・オペレーションが前記N個のベクトルの少なくとも1つにおけるK個のエレメントすべてに対して実質的に同時に前記K個の演算装置によって遂行されるものと、

前記所定のサイクル・タイム後の次のサイクルタイムにおいて他のベクトル・オペレーションを遂行するための手段と、

を含む計算機プロセッサ。

【請求項2】前記N個のレジスタのセットにおける各レジスタはM個のビット記憶ロケーション（但し、M＞0）を有することを特徴とする請求項1に記載の計算機プロセッサ。

【請求項3】前記ベクトル・オペレーションを遂行するための手段である第1演算装置及び前記他のベクトル・オペレーションを遂行するための手段である第2演算装置は同じ演算装置であることを特徴とする請求項1に記載の計算機プロセッサ。

【請求項4】マトリクス制御装置を含むことを特徴とする請求項1に記載の計算機プロセッサ。

【請求項5】前記マトリクス制御装置を前記K個の処理エレメントに接続するコマンド・バスを含むことを特徴とする請求項4に記載の計算機プロセッサ。

【請求項6】前記マトリクス制御装置を前記K個の処理エレメントに接続する少なくとも1つのデータ・バスを含むことを特徴とする請求項4又は5に記載の計算機プロセッサ。

【請求項7】前記少なくとも1つのデータ・バスは前記K個の処理エレメントのうちの2個以上の処理エレメントによって共用されることを特徴とする請求項5に記載の計算機プロセッサ。

【請求項8】サイクル・タイムを有する計算機プロセッサにして、

マトリクス制御装置と、

前記サイクル・タイムのうちの先行のベクトル・オペレーションの1サイクルにおいて後続のベクトル・オペレーションを遂行するための手段と、

を含む計算機プロセッサ。

2

【請求項9】前記手段はベクトル演算オペレーションを遂行することを特徴とする請求項8に記載の計算機プロセッサ。

【請求項10】前記マトリクス制御装置を前記K個の処理エレメントに接続するコマンド・バスと、

前記マトリクス制御装置を前記K個の処理エレメントに接続する少なくとも1つのデータ・バスと、

を含むことを特徴とする請求項8に記載の計算機プロセッサ。

【請求項11】サイクル・タイムを有する計算機システムにして、

メイン・メモリと、

キャッシュ・メモリと、

記憶制御装置と、

中央処理装置と、

マトリクス処理装置と、

前記中央処理装置から前記マトリクス処理装置に命令を転送するためのマトリクス命令バスと、

前記キャッシュ・メモリから前記マトリクス処理装置にデータを転送するための中央処理装置データ・バスと、

前記中央処理装置は前記記憶制御装置と前記キャッシュ・メモリとを介して前記メイン・メモリに接続する手段と、

前記マトリクス制御装置は前記キャッシュ・メモリを介して前記メイン・メモリに接続する手段と、

前記記憶制御装置はメモリ・アドレス・バスによって前記メイン・メモリに接続する手段と、

前記記憶制御装置は第1データ・バスによって前記中央処理装置に接続する手段と、

前記キャッシュ・メモリは第2データ・バスによって前記マトリクス処理装置に接続する手段と、

前記マトリクス処理装置は前記サイクル・タイムのうちの先行のベクトル・オペレーションの1サイクルにおいて後続のベクトル・オペレーションを遂行するための手段を含むことと、

K個の処理エレメント（但し、K＞1）にして、前記K個の処理エレメントの各々はN個のレジスタのセット（但し、N＞1）及び演算装置を含み、前記レジスタの各々のI番目のエレメントはベクトル長KのI番目のベクトルのための記憶ロケーションを提供し（但し、1≦I≦N）、それによって、N個のベクトルを提供するものと、

前記N個のベクトルの少なくとも1つの間でベクトル・オペレーションを所定のサイクル・タイムで遂行し、前記ベクトル・オペレーションの結果を前記N個のベクトルの1つに記憶するための手段にして、前記ベクトル・オペレーションが前記N個のベクトルの少なくとも1つにおけるK個のエレメントすべてに対して実質的に同時に前記K個の演算装置によって遂行されるものと、

前記所定のサイクル・タイム後の次のサイクルタイムに

3

おいて他のベクトル・オペレーションを遂行するための手段と、
を含む計算機システム。
【請求項１２】サイクル・タイムを有する計算機システムにして、
メイン・メモリと、
キャッシュ・メモリと、
記憶制御装置と、
中央処理装置と、
マトリクス処理装置と、
前記中央処理装置から前記マトリクス処理装置に命令を転送するためのマトリクス命令パスと、
前記キャッシュ・メモリから前記マトリクス処理装置にデータを転送するための中央処理装置データ・パスと、
を含み、
前記中央処理装置は前記記憶制御装置と前記キャッシュ・メモリとを介して前記メイン・メモリに接続する手段と、
前記マトリクス制御装置は前記キャッシュ・メモリを介して前記メイン・メモリに接続する手段と、
前記記憶制御装置はメモリ・アドレス・バスによって前記メイン・メモリに接続する手段と、
前記記憶制御装置は第１データ・バスによって前記中央処理装置に接続する手段と、
前記キャッシュ・メモリは第２データ・バスによって前記マトリクス処理装置に接続する手段と、
前記マトリクス処理装置は、
Ｋ個の処理エレメント（但し、Ｋ＞１）にして、前記Ｋ個の処理エレメントの各々はＮ個のレジスタのセット（但し、Ｎ＞１）及び演算装置を含み、前記レジスタの各々のＩ番目のエレメントはベクトル長ＫのＩ番目のベクトルのための記憶ロケーションを提供し（但し、１≦Ｉ≦Ｎ）、それによって、Ｎ個のベクトルを提供するものと、
前記Ｎ個のベクトルの少なくとも１つの間でベクトル・オペレーションを所定のサイクル・タイムで遂行し、前記ベクトル・オペレーションの結果を前記Ｎ個のベクトルの１つに記憶するための手段にして、前記ベクトル・オペレーションが前記Ｎ個のベクトルの少なくとも１つにおけるＫ個のエレメントすべてに対して実質的に同時に前記Ｋ個の演算装置によって遂行されるものと、
前記所定のサイクル・タイム後の次のサイクルタイムにおいて他のベクトル・オペレーションを遂行するための手段と、
を含むことを、
特徴とする計算機システム。
【請求項１３】マトリクス制御装置と、
前記マトリクス制御装置を前記Ｋ個の処理エレメントに接続するためのコマンド・バスと、
前記マトリクス制御装置を前記Ｋ個の処理エレメントに

4

接続するための少なくとも１つのデータ・バスと、
を含むことを特徴とする請求項１２に記載の計算機システム。
【請求項１４】サイクル・タイムを有する計算機システムにして、
メイン・メモリと、
中央処理装置と、
マトリクス処理装置と、
前記中央処理装置から前記マトリクス処理装置に命令を転送するためのマトリクス命令パスと、
前記メイン・メモリから前記マトリクス処理装置にデータを転送するための中央処理装置データ・パスと、
前記中央処理装置は第１データ・バスによって前記メイン・メモリに接続する手段と、
前記マトリクス制御装置は第２データ・バスによって前記メイン・メモリに接続する手段と、
前記記憶制御装置はメモリ・アドレス・バスによって前記メイン・メモリに接続する手段と、
前記マトリクス処理装置は前記サイクル・タイムのうちの先行のベクトル・オペレーションの１サイクルにおいて後続のベクトル・オペレーションを遂行するための手段を含むことと、
Ｋ個の処理エレメント（但し、Ｋ＞１）にして、前記Ｋ個の処理エレメントの各々はＮ個のレジスタのセット（但し、Ｎ＞１）及び演算装置を含み、前記レジスタの各々のＩ番目のエレメントはベクトル長ＫのＩ番目のベクトルのための記憶ロケーションを提供し（但し、１≦Ｉ≦Ｎ）、それによって、Ｎ個のベクトルを提供するものと、
前記Ｎ個のベクトルの少なくとも１つの間でベクトル・オペレーションを所定のサイクル・タイムで遂行し、前記ベクトル・オペレーションの結果を前記Ｎ個のベクトルの１つに記憶するための手段にして、前記ベクトル・オペレーションが前記Ｎ個のベクトルの少なくとも１つにおけるＫ個のエレメントすべてに対して実質的に同時に前記Ｋ個の演算装置によって遂行されるものと、
前記所定のサイクル・タイム後の次のサイクルタイムにおいて他のベクトル・オペレーションを遂行するための手段と、
を含む計算機システム。
【発明の詳細な説明】
【０００１】
【産業上の利用分野】本発明は、概して云えば、データ処理システムの分野に関するものであり、詳しく云えば、本発明は、複数個のプロセッサにおいて数値集中計算(numericallyintensive computing) するためのデータ処理システムに関するものである。更に詳しく云えば、本発明は、Ｎ個のプロセッサがＮ個のエレメント・ベクトル上で縮小命令セットの命令を同時に実行する多重プロセッサ・システムに関するものである。

5

【0002】

【従来の技術】コンピュータにおける計算力は成長してきたし、急速に成長し続けている。この増大した計算力は、新たな方法でコンピュータを使用するための新たな機会をその計算力のユーザに与えており、従って、コンピュータ・パワーを増大させるための更なる要求を生み出しいる。増大した計算力が特に有用である1つの領域は、数値集中計算の領域である。数値集中計算は、大量のデータに関する限られたオペレーション・セットの計算に関連している。数値集中計算の技法は、イメージ処理、信号処理、人工知能、或いは気象力学又は流体力学のコンピュータ・シミュレーションのような幅広い分野の適用業務において使用される。これらの適用業務における数値集中計算の技法が直面する共通の問題は、各適用業務と関連した大量のデータに関して必要な有限の数の関数を、可能な限りの最小の時間で計算することである。

【0003】初期の従来技術のコンピュータ・システムは、中央処理装置（CPU），メモリ、及び入出力（I／O）コントローラより成るものであった。CPUは、データ又は命令をメモリに転送するように或いはそれらをメモリから取り出すようにI／Oコントローラに命令するプログラム、及び種々の命令セットに従ってデータを処理するプログラムを実行してした。データを処理することは、CPUが遂行すべき特殊な命令をデコードするようなオペレーション、その特殊な命令を実行するようなオペレーション、及びその結果をメモリに戻すようなオペレーションを含んでいた。CPUは、それがプログラムにおける命令のリストからの命令を一時に1つずつ処理するという点では複雑ではなかった。しかし、このタイプのCPUは、それが一時に1つの命令しか処理しなかったので遅かった。その遅い処理時間に応答して、従来技術のシステムは、プロセッサをパイプライン化することを導入した。

【0004】パイプライン化したプロセッサでは、異なる命令に対して、CPUの種々の独立した機能が一時に生じる。例えば、1つの命令に対するデータを取り出すためのメモリに対するプロセッサの命令は、プロセッサが他の命令のオペレーション・コードをデコードしている時に生じようとしているかもしれない。パイプライン化することは、何れの個々の命令の実行もスピード・アップするものではないが、それは、前の命令が処理を完了してしまう前に後続の命令がCPUにおける処理を開始するため、一連の命令の処理をスピード・アップしている。パイプライン化に加えて、従来技術は、限定された命令セットが数値的問題に対して反復的に使用されることも認識してした。その結果、従来技術は、それら限定された命令セットを素早く実行するために、CPU内に特殊プロセッサの機能的ブロックを導入した。例えば、いくつかのCPUは、加算、乗算、又はブランチだけを遂行するための機能的ブロックを含んでいた。これら機能的ブロックは、メインCPUよりもずっと速くこれらの限定された機能を実行したし、これは数値的関数を処理する速度も増加させた。

6

【0005】パイプライン化又は限定された命令のための別個の機能ユニット、或いはそれらの両方を組み込んだ従来技術のプロセッサでは、単一の命令ストリームが単一のデータ・ストリーム上で動作していた。即ち、各命令は、定義されたデータに関して一時に1つの計算を行うように動作してした。そのようなプロセッサは、単一命令・単一データ処理の略称であるSISDと呼ばれる。SISDタイプのプロセッサの問題は、プログラムのセグメントが多数の多様なエレメントに適用されることをそのプログラムが必要とする場合、そのプログラムは、順次に何回もそのセグメントを循環しなければならないことである。これは、多くのタイプの数値的問題に対する状況であった。そして、このようなタイプの問題に関するプロセッサ速度の増加を援助するために、SIMD（単一命令・多重データ処理）タイプのプロセッサが開発された。

【0006】SIMDプロセッサ・ユニットは、同じ機能的能力を持った複数個の処理エレメントより成り、そしてそれら処理エレメントは、それぞれ、各処理エレメントが別個のデータ・ストリームに関して動作する時、一時に1つの命令基づいて動作する。代表的なSIMDプロセッサ・ユニットは、3つの主要な素子、即ち、処理エレメントのアレイ、経路指定ネットワーク、及びコントローラを含む。対照的に、本発明によるプロセッサは経路指定ネットワークを使用せず、その結果、低コストのプロセッサが得られ、メイン・メモリ及び処理エレメントに及びそれらからデータを移動させるための時間の減少が得られる。経路指定ネットワークは、任意の1つの処理エレメントからの結果を、それを必要とするそのアレイ内の他の任意の処理エレメントにコミュニケートする。コントローラは、そのSIMDプロセッサ・ユニットが一部分となるメインCPUからの命令及びデータに応答して、その処理エレメントのアレイ及び経路指定ネットワークのオペレーションを制御する。

【0007】マルチプロセッサSIMDタイプのシステムによる問題は、コンピュータ・メモリ・システムからデータを十分に速い速度で得るためのマルチプロセッサ・アレイの能力と多数のプロセッサの使用を最適化するためのメインCPUの能力とによって、システム・パフォーマンスが制限されるということである。とりわけ、メイン・メモリからのデータをアレイにおける各プロセッサに供給するメモリ・システムを構築することは、非常に高いメモリ帯域幅（BW）を必要とし、従って、余りにも高価であり、或いは殆どのアプリケーションにとって非実用的である。更に、個々の各プロセッサがそのアレイにおける他のプロセッサと直接にコミュニケート

7

する経路指定ネットワークは、そのプロセッサのアレイの全体的なパフォーマンスを下げる。その従来技術は、限られた成功でもってこれらの問題を解決しようとした。更に詳しく言えば、その従来技術は、プロセッサ・アレイにおけるそれら複数個のプロセッサをマトリクス状に接続し、それらプロセッサがそれらのいくつかの最も近接したプロセッサとだけコミュニケートするようにした。

【０００８】この配列は、最も近接したプロセッサがコミュニケートする能力を増大させ、従って、命令がそれら最も近接したプロセッサ相互間だけのコミュニケーションを必要とする時にはそれらプロセッサの全体的なパフォーマンスを増大させる。しかし、殆どの命令は無作為なプロセッサ相互間のコミュニケーションを必要とするものであり、最も近接したプロセッサを介した無作為なプロセッサへの経路指定コミュニケーションは、そのプロセッサ・アレイの全体的なパフォーマンスを減少させる。その結果、プロセッサの近接プロセッサ接続は、実行される命令の数及びタイプによっては、実際に、マルチプロセッサ・アレイのパフォーマンスを減少させる。各プロセッサが多数の他のプロセッサに直接に接続しているハイパーキューブ・タイプの配列に複数のプロセッサが接続させる時、それらプロセッサ相互間のコミュニケーションの速度は増加する。このタイプの配列によって、直接に相互接続されてないプロセッサ相互間のコミュニケーションを必要とすることが少なくなり、そのプロセッサ・アレイのパフォーマンスが増加するであろう。しかし、プロセッサ・アレイ・パフォーマンスの増加は、依然として、そのプロセッサ・アレイと関連したメモリ帯域幅によって制限される。プロセッサ・アレイは、データを個々のプロセッサに配送し得る速さで命令を実行することはできる。ハイパーキューブ配列は、そのようなメモリ・システムをそのハイパーキューブ配列と結合して実施することが多くのコンピュータ・システムにおいて非実用的であるような高いメモリ帯域幅を必要とする。

【０００９】最近の高パフォーマンスのコンピュータにとって、パフォーマンスは主としてその機械のサイクル・タイムによって制限される。全体的なパフォーマンスを増加させるために、しばしば、並列化が使用される。多数のプロセッサと共に共用メモリの並列システムを構築することが非常に困難であることはわかっている。最近、いくつもの分散型メモリ並列システムが構築されている。Ｉｎｔｅｌ　Ｐａｒａｇｏｎ、Ｃｒａｙ　Ｔ３－Ｄ、ＩＢＭ　ＳＰ１はこのクラスの機械の例である。一般には、多数のプロセッサでもって分散型メモリ・システムを最適に利用すること及びそれのソフトウエアを書くことは更に困難である。これらの機械は、しばしば、通信障害によって遅滞させられる。又、殆どのシステム・コンポーネントの複製及び通信ネットワークの付加的

8

コスト及びオーバヘッドのために、これらの高度な並列機械の価格性能比は、基礎的なユニプロセッサに対する価格性能比ほど有利なものではないことが多い。

【００１０】本発明によるコンピュータ・システムにおける２つの主要な考察点は、命令の実行の制御とメモリ管理である。制御の考察点及びオペレーティング・システムの観点から云えば、それは、単一の命令ストリームしか実行されない場合、又はメモリの１つのイメージしか維持されない場合、大変望ましい。換言すれば、オペレーティング・システムの観点から云えば、その機械が単一のＣＰＵの機械のようにみえることが望ましい。本発明は、「単一の命令ストリーム及びメモリ・マップ」モデルの傘の下で同時に動作しながら、或クラスのＮＩＣ（数値集中計算）問題に関して非常に高いパフォーマンスを得るシステム構成を定義するためのものである。

【００１１】最近の高パフォーマンス・コンピュータにとって、パフォーマンスは、メモリ・システムから得られる帯域幅によっても制限される。更に、本発明は、或クラスの数値集中計算の問題に対するメモリ帯域幅の要件を少なくするために、上記のシステム構造で働くアルゴリズム構造を提供する。

【００１２】問題の数値的強度を数値化する１つの簡単な測定法は、浮動小数点演算（フロップス）の合計数をカウントすること、及びその計算に関連したデータ・ポイントの合計数（又は、必要なメモリ・スペースの量）によってそれを除することである。この比率を「浮動小数点演算対データ比」と呼ぶことにする。要するに、この比をＦＤＲと表すことにする。ＦＤＲはその問題に対して全体として計算可能であり、その問題によって包含される副次問題に対しても計算可能である。一般的には、副次問題に対するＦＤＲは、その問題全体に対するＦＤＲよりもかなり小さい。初歩的なスカラ演算レベルでは、ＦＤＲは１よりも小さいであろう。殆どのＮＩＣ問題に対して、当初の問題を適当にブロック化（アルゴリズム的に）して、種々なレベルのメモリ階層に適合した副次問題にすることによって、ＦＤＲはかなり改良可能である。

【００１３】一般的には、より大きなメモリ・スペースが所与のレベルにおいて利用可能にされる場合、より高い値のＦＤＲが達成可能である。階層メモリ・システムでは、所与のレベルのメモリにおけるメモリ帯域幅（ＢＷ）の要件は、そのステージにおけるＦＤＲ比に逆比例する。最高のレベル（最大のメモリ・サイズ）では、帯域幅要件は最も小さく、そしてより低い（及びより小さい）メモリのレベルに進むにつれて、帯域幅要件は次第に増大する。その正確な数は、その問題、使用されるアルゴリズム、及び利用可能な計算資源の量に依存するであろう。一般的には、計算資源が高くなるほど、帯域幅要件も高くなると言える。以下では、メモリの帯域幅要件を大きく増大させることなく、大量の計算資源が最適

に利用可能であるというアルゴリズム的な制限のない例を示す。一般には、アルゴリズムは種々なレベルのメモリ及びそれらの関連した帯域幅を最適に利用するように適合可能である。

【００１４】

【発明が解決しようとする課題】本発明の目的は、改良された計算機システムを提供することにある。

【００１５】本発明のもう１つの目的は、数値集中計算の問題に対する改良された計算機システムを提供することにある。

【００１６】本発明の更にもう１つの問題は、多数のプロセッサを持った改良された計算機システムを提供することにある。

【００１７】本発明の更にもう１つの目的は、制限された数の命令をより速く実行するための改良された計算機システムを提供することにある。

【００１８】本発明の更にもう１つの目的は、計算機システムのメモリ帯域幅を大きく増大させることなく、制限された数の命令をより速く実行するための改良された計算機システムを提供することにある。

【００１９】本発明の更にもう１つの目的は、制限された数の命令をより速く実行するための改良された廉価の計算機システムを提供することにある。

【００２０】

【課題を解決するための手段】本発明の幅広く見た場合の視点は、計算機システムのスループットを大幅に増大させるためにデータ・ワード対のグループに関するオペレーションを同時に遂行する並列プロセッサを持った計算機システムである。

【００２１】本発明の幅広く見た場合のもう１つの視点は、先行のベクトル・オペレーションの１サイクル・タイム内で後続のベクトル・オペレーションを遂行する計算機システムである。

【００２２】本発明の更に特殊な視点は、プロセッサ・サイクル・タイムを持った計算機プロセッサである。計算機プロセッサは、Ｋ個の処理エレメント（但し、Ｋ＞１）を有する。それらＫ個のエレメントの各々は、Ｎ個のレジスタのセット（但し、Ｎ＞１）及び演算装置を有する。それらレジスタの各々のＩ番目のエレメントは、ベクトル長ＫのＩ番目のベクトル・レジスタに対する記憶ロケーションを与える。その計算機プロセッサは、Ｎ個のベクトルのうちの少なくとも１つにおける又はそれらの間のベクトル演算オペレーションを所定のサイクル・タイムで遂行するための手段及びＮ個のベクトルのうちの１つにおけるオペレーションの結果を記憶するための手段を有する。この場合、ベクトル演算オペレーションは、Ｎ個のベクトルのうちの少なくとも１つについて、Ｋ個のエレメントすべてに対するＫ個の演算装置によって実質的に同時に遂行される。その計算機プロセッサは、所定のサイクル・タイム後の次のサイクルタイムにおいてもう１つのベクトル・オペレーションを遂行するための手段を有する。



【００２３】

【実施例】メモリ帯域幅（ＢＷ）の問題を処理した後、次の問題は、その機械の所与のサイクル・タイムに対して大量の計算を行い得るような計算機そのものの編成である。パフォーマンスの１つの測定法は、１サイクル当たりの浮動小数点演算（ＦＰＣ）である。ＩＢＭ　Ｓ／３９０のような伝統的なスカラ・アーキテクチャに対しては、１つの浮動小数点演算を発生するためにいくつかのサイクルが必要とされる。先ず、オペランドをレジスタ（又は、浮動小数点装置）にフェッチする必要があり、これにパイプライン化した演算が続き、しかる後、その結果がメイン・メモリに戻されて記憶される。そのようなシステムに対するＦＰＣは、典型的には、わずかな端数である。ＩＢＭ　ＲＳ／６０００のような機械は多数の機能的ユニットを有し、その各々は調整された態様で１つの指定された機能を行い、その結果、理想的な環境下では１サイクル当たり２又はそれ以上の浮動小数点演算を生じる。ベクトル機械も、長いベクトルにおけるパイプライン待ち時間を隠蔽することによって同様の結果を得る。

【００２４】伝統的なベクトル・アーキテクチャでは、同じオペレーションが、独立したオペランドのセットに関して遂行される。独立したオペランドの数はベクトル長と呼ばれる。現在のベクトル機械では、これらオペランドは、パイプライン化した演算装置によって逐次に処理される。或機械では、スループット（ＦＰＣ）を増加させるために、演算パイプと呼ばれる多数の演算装置が設けられる。一般的には、多くのパイプが加えられると、メモリ帯域幅もそれに従って増加する。

【００２５】本発明は、オペランドの各セット（そのベクトルの各エレメント）が独立した演算装置によって処理されるシステム構造である。その場合、利用可能な演算装置の数はベクトル長に等しい。Ｓ／３９０の用語では、それはベクトル・セクション・サイズ（ＶＳＳ又はＫ）と呼ばれる。ベクトル実行のために必要なサイクルの数はパイプライン長に等しくなるであろう。１つのパイプでは、加算のようなオペレーションは、典型的には、２又はそれ以上のステップに分割される。その場合、各ステップは１機械サイクルで行われる。パイプライン長の遅れは、そのパイプへの入力とそのオペレーションの最終結果を伴うパイプの出力との間の機械サイクルにおける遅れである。しかし、演算装置のパイプラインの性質のために、適当な命令スケジューリングによって、各サイクルで「ベクトル・オペランド」のセットを送りそして「ベクトル結果」得ることが可能でなければならない。これは、ＳＩＭＤ（単一命令・多重データ処理）並列処理の１つの形式である。しかし、それは、米国のＴｈｉｎｋｉｎｇ　Ｍａｃｈｉｎｅ社のＣＭ－２０

11

0、CM－5等のような伝統的なSIMD機械とは非常に異なっている。そのようなアーキテクチャにとって、得ることのできる1サイクル当たりの最大の浮動小数点演算は2＊VSS（乗算・加算オペレーションが2浮動小数点演算としてカウントされる）である。更に、そのパフォーマンスのレベルはパラメータVSS又はKによってパラメータ化される。理論的には、これは達成可能な最良のパフォーマンスである。個のレベルのパフォーマンスを実際に得ることは、データを送ること及び演算装置からの結果を非常に高い速度で記憶することを必要とする。これを次に説明する。

【0026】伝統的なベクトル・アーキテクチャでは、オペランドを一組のベクトル・レジスタ又はメモリから得ることができる。RISCスカラ・アーキテクチャでは、すべてのオペランドが、先ず、一組のスカラ・レジスタにロードされる。演算装置は、すべてのオペランドをレジスタから取得し、すべての結果をレジスタに戻して記憶する。これは、演算装置に対するメモリへの又はメモリからの2ステップ・データ・パスである。本発明の好適な実施例では、処理エレメントはRISCベクトル・アーキテクチャを使用する。そのアーキテクチャでは、すべての演算装置がベクトル・レジスタとだけ対話し（それらのオペランドをベクトル・レジスタのみから取得し、それらの結果をベクトル・レジスタのみに記憶し）、一方、ベクトル・レジスタはメモリ・システムと対話する。

【0027】RISCアーキテクチャは、1990年1月発行の「IBM Journal of Research and Development」第34巻、第1号の記事に記載されている。これは他のレベルの減結合を直接に提案している。演算装置は、その演算装置と同じインデックス番号を持ったベクトル・レジスタのエレメントだけに対話する必要がある。便宜上、演算装置に0からK－1までの番号を付することにする。N（もう1つのパラメータ）個のベクトル・レジスタがあるものと仮定する。その場合、演算装置は、N個のスカラ・レジスタとだけ対話する必要がある。最良の方法では、これらのレジスタ及び演算装置は同じチップ上にあるものと考えられる。演算装置とレジスタとのこの組合せを処理エレメント（PE）と呼ぶことにする（エレメントの合計数＝N＊K）。N個のベクトル・レジスタがK個のPE上に分配される。このK個のPEの集合体は、NICA（数値集中計算アクセレレータ）と呼ばれる。データがNICAレジスタにある場合、それは1サイクル当たり2＊K浮動小数点演算のピーク・パフォーマンスを得ることができる。

【0028】図1は、本発明に従ってマトリクス装置（MXU）（或いは、ベクトル・プロセッサ）を使用したコンピュータ・システムの1つの実施例を示す。このコンピュータ・システムでは、中央処理装置（CPU）45及びMXU50が、記憶制御装置（SCU）25及びキャッシュ・バッファ・メモリ（L2キャッシュ）35を介してメイン・メモリ（メモリ・システム）10に接続される。CPU45は、MXU命令パス43、MXUアドレス及びカウント・パス42、及びMXU記憶リクエストIDパス41を介してMXU50に接続される。SCU25は、メモリ・アドレス・パス15によってメイン・メモリ10に接続され、アドレス・パス47によってCPU45に接続される。L2キャッシュ35は、メモリ・データ・パス20によってメイン・メモリ10に接続される。更に、そのL2キャッシュ35は、CPUデータ・パス37によってCPU45に接続され、MXUデータ・パス39によってMXU50に接続される。CPU45は、ブランチ装置（BRU）57、浮動小数点装置（FPU）48、固定小数点装置（FXU）49、レベル1命令キャッシュ装置（IU）46、及びレベル1データ・キャッシュ装置（DU）44より成る汎用プロセッサである。そのIU46及びDU44は任意選択のものであり、単にパフォーマンス上の理由で使用されるだけである。

12

【0029】BRU57は、CPU45及びMXU50に対するコントロール・フロー（次に実行されるべき命令を決定する）を与える。BRU57は、実行されるべき次の命令のメモリ・ロケーションのアドレスを発生する。その命令がIU46に存在する（それが最近アクセスされた）場合、その命令はIU46からフェッチされる。その命令がIU46に存在しない場合、BRU57はそのロケーションに対するリクエストを発生し、L2キャッシュ・アドレス・パス47を介してSCU25にそのリクエストを送る。SCU25は、L2キャッシュ35の内容のディレクトリを持っている。そのリクエストされたメモリ・ロケーションがL2キャッシュ35に存在する場合、SCU25は、L2キャッシュ35におけるそのロケーションをアクセスするに必要なコントロールを発生し、L2コントロール・パス26を介してそのL2キャッシュにアクセス・コントロールを送る。L2キャッシュ35はそのアクセスを遂行し、CPUデータ・パス37を介してCPU45にそのロケーションの内容を送る。リクエストされたアドレスがL2キャッシュ35に存在しない場合、SCU25はメモリ・アクセス・リクエストを発生し、メモリ・アドレス・パス15を介してメモリ・システム10にそのリクエストを送る。メモリ・システム10はそのリクエストされたロケーションをアクセスし、メモリ・データ・パス20を介してL2キャッシュ35にそのリクエストされたロケーションの内容を戻す。L2キャッシュ35は、CPUデータ・パス37を介してCPU45にそのリクエストされたメモリ・ロケーションの内容を戻す。IU46は、そのリクエストされたメモリ・ロケーションの内容を受け取ると、ディスパッチ及び実行のための命令をBRU57に送る。

13

【００３０】ＢＲＵ５７は実行されるべき次の命令を調べ、その命令を適当な実行装置にディスパッチする。ブランチ命令は、ＢＲＵ５７によって実行される。スカラ浮動小数点命令は、ＦＰＵ４８によって実行される。ベクトル命令は、ＭＸＵ５０によって実行される。メモリ・アクセス命令は、ＦＸＵ４８及び関連の実行装置により共同して実行される。その場合、ＦＸＵ４８はロード命令又はストア命令のアドレスを発生し、ＤＵ４４に（ＦＸＵ及びＦＰＵメモリ・アクセスのために）又はＳＣＵ２５に（ＭＸＵメモリ・アクセスのために）リクエストを送る。ベクトル命令は、ＢＲＵ５７によって、ＭＸＵ命令パス４１を介してＭＸＵ５０にディスパッチされる。ベクトル・メモリ・アクセス命令は、ＭＸＵ５０及びＦＸＵ４９の両方にディスパッチされる。

【００３１】上述のように、ベクトル・メモリ・アクセス命令は、ＦＸＵ４９及びＭＸＵ５０によって共同して実行される。ＦＸＵ４９は、メモリ・アクセスのアドレスを計算し、メモリ・リクエストをＤＵ４４に送る。ＤＵ４４は、リクエストされたロケーションがＬ１データ・キャッシュに存在しないことを確認するためにキャッシュ・ディレクトリ・ルックアップを行う。そのメモリ・ロケーションが存在する場合、そのロケーションは、先ず、Ｌ２キャッシュ３５にフラッシュ・アウトされてそれをＭＸＵメモリ・アクセスにとって明白なものにする。そこで、ＤＵ４４はメモリ・アクセス・リクエストをＳＣＵ２５に送る。命令メモリ・アクセスと同様に、ＳＣＵ２５は、そのリクエストされたメモリ・ロケーションがＬ２キャッシュ３５に存在するかどうかを決定するためにＬ２キャッシュ・ディレクトリを調べる。そのロケーションがＬ２キャッシュ３５に存在しない場合、メモリ・システム１０におけるそのメモリ・ロケーションがリクエストされ、その内容がメモリ・データ・パス２０を介して戻され、Ｌ２キャッシュ３５に置かれる。

【００３２】ＭＸＵロード命令に対しては、そのリクエストされたメモリ・ロケーションの内容が、ＭＸＵデータ・パス３９を介してＭＸＵ５０に戻される。ＳＣＵ２５は、ロード・データが、ＭＸＵ－Ｌ２キャッシュ・コントロール・パス５４を使用してＭＸＵデータ・パス３９上に得られることを、ＭＸＵ５０に知らせる。ＭＸＵストア命令に対しては、ＳＣＵ２５及びＬ２キャッシュ３５は、そのストア命令のデータがＭＸＵ５０によってＭＸＵデータ・パス３９上に存在するまで待つ。その時点で、そのデータはＬ２キャッシュ３５におけるそのアドレスされたメモリ・ロケーションに置かれる。マトリクス制御装置（ＭＸＣＵ）５１は、ストア・データがＭＸＵ－Ｌ２キャッシュ・コントロール・パス５４を使用してＭＸＵデータ・パス３９上に得られることをＳＣＵ２５に知らせる。

【００３３】ＭＸＵ命令パス４１を介してディスパッチされるベクトル命令に対して、マトリクス制御装置（ＭＸＣＵ）５１は実行されるべき命令を受け取り、そしてその後の実行のためにその命令を命令待ち行列に置く。すべての必要な制御情報が受け取られる時、（メモリ・アクセス命令が、ＦＸＵ４９からのデータ位置合わせ及びエレメント・カウントのために下位アドレス・ビットを使用し）ＭＸＣＵ５１において命令の実行が始まる。エレメント・カウントはマトリクス・カウント・レジスタ（ＭＣＲ）５３に保持され、ベクトル・メモリ・アクセス命令によってロード又は記憶されるべきエレメントの数を指定する。そこから、ＭＸＣＵ５１は適当なコマンド（元の命令のフォーマット化バージョン）及び処理エレメント・マスク（命令を実行する場合に、どのＭＸＰＥが参加すべきかを識別する）を設定し、ＭＸＰＥコマンド・パス７０を介してマトリクス処理エレメントのアレイ（ＭＸＰＥ１－ＭＸＰＥＫ）にそのコマンド（又は、一連のコマンド）をディスパッチして実行させる。

【００３４】ベクトル・ロード命令に対しては、ＭＸＣＵ５１は、Ｌ２キャッシュ３５からＭＸＵデータ・パス３９を介して当該データを受け取り、そのデータを適当な目標ＭＸＰＥデータ・パスへ経路指定してその目標ＭＸＰＥに送る。ベクトル・ストア命令に対しては、ＭＸＣＵ５１は、ソースＭＸＰＥからそれぞれのＭＸＰＥデータ・パスを介して当該記憶データを受け取り、その記憶データを適当に位置合わせするように経路指定し、そしてＭＸＵデータ・パス３９を介してＬ２キャッシュ３５にそのデータを送る。ベクトル演算命令に対しては、ＭＸＣＵ５１は、ＭＸＰＥからＭＸＰＥ状態パス７２を介して結果状態（表示を除く）を受け取る。この状態は、マトリクス状態及び制御レジスタ（ＭＸＳＣＲ）５２において捕捉され、マスク可能例外条件が存在する場合に、ＭＸＵ状態パス４２を介してＢＲＵ５７に割り込み要求の形で送られる。

【００３５】ＭＸＰＥはすべて、ＭＸＰＥコマンド・パス７０を介してコマンドを受け取る。ＭＸＣＵは、各コマンドに従ってマスクを施し、どのＭＸＰＥが命令の実行に参加すべきかを識別する。ＭＸＰＥのサブセットは、ＭＸＰＥの数よりも少ないエレメント・カウントのような多数のファクタ、位置合わせされてないメモリ・アクセスによって選択可能である。即ち、プログラムは、ＭＸＳＣＵ５２の開始及び終了範囲フィールドにおいてサブセットを明瞭に指定する。マスク・オフされたＭＸＰＥはその命令を廃棄する。マスク・オフされてないＭＸＰＥは実行のための命令を受け取り、待ち行列に入れる。ベクトル・ロード命令に対しては、選択されたＭＸＰＥに対するロード・データが、ロード・コマンドによって、それぞれのＭＸＰＥデータ・パスを介して受け取られる。そこで、選択されたＭＸＰＥはロード・データを目標レジスタにロードする。ベクトル・ストア命令に対しては、選択された各ＭＸＰＥはソース・レジスタをアクセスし、それのＭＸＰＥデータ・パスを介して

15

そのデータを与える。ベクトル演算命令に対しては、選択された各ＭＸＰＥはそのリクエストされた演算機能をＳＩＭＤ態様で遂行し、ソース・オペランドをマトリクス・レジスタ（ＭＸＲ）から取得し、そしてその結果を目標ＭＸＲの中に置く。演算オペレーションに対する状態は、選択されたＭＸＰＥのマトリクス条件レジスタの各々の中に置き、そしてその生じた例外事項の概略がＭＸＰＥ状態バス７２を介してＭＸＣＵ５１に送られる。

【００３６】図５は、ＭＸＣＵ５１の編成の更に詳細な図を示す。ＭＸＣＵ５１は、３つの主要なコンポーネント、即ち、命令制御装置（ＩＣＵ）５７０、ロード・データ装置（ＬＤＵ）５６０、及びストア・データ装置（ＳＤＵ）５５０より成る。ＩＣＵ５７０は、命令待ち行列装置５７１、コマンド発生装置５７２、及びコマンド・ディスパッチ装置５７３より成る。命令待ち行列装置５７１は、命令に対するすべての前提制御情報を受け取ってしまうような時間まで、ＣＰＵ４５からディスパッチされるＭＸＵ命令をバッファするために使用される。そのような前提データの例は、如何に多くのメモリ・アクセスがＣＰＵ４５によって設定されようとしているかをＭＸＵ５０に表示するロード命令及びストア命令のためのエレメント・カウントである。一旦、すべての必要な前提データが収集されてしまうと、その命令はコマンド発生装置５７２に進む。コマンド発生装置５７２は当該命令及び関連の制御情報を評価し、そしてＭＸＰＥにディスパッチされるべきその指定された機能を適切に且つ正しく実行する適当なコマンドを設定する。そのような設定は、ＭＸＳＣＲ５２の開始及び終了範囲設定に基づくＭＸＰＥマスクの生成、又はメモリ・アクセス情報のアドレス位置合わせ及びエレメント・カウントを含む。何れの特別のレジスタ・アドレシング・モードもこのステージにおいて実施可能である。コマンド・ディスパッチ装置５７３はＭＸＵ５０の現在の状態を評価し、次のコマンドがＭＸＰＥにディスパッチ可能であるかどうかを決定する。レジスタ従属性が可能となる場合の順序外ディスパッチのようなパフォーマンス強化技法がこのステージにおいて使用可能である。

【００３７】ＬＤＵ５６０は、ロード命令の実行準備ができる前に到達したＭＸＵロード・データをデータ・バッファ５０４に与える。ここでは、ロード・データの順序外復帰をサポートするようなパフォーマンス強化技法がサポート可能である。そこで、ＬＤＵ５６０は、任意のアービトラリＭＸＰＥデータ・バス・ポートへのロード・データの経路指定をサポートすることを意図したマルチプレクサのネットワーク、即ち、ルータ５０８を提供する。ＳＤＵ５５０は、任意のアービトラリＭＸＰＥデータ・バス・ポートからのストア・データの経路指定をサポートすることを意図したマルチプレクサのネットワーク、即ち、ルータ５３４を提供する。ＭＸＰＥデータ・バス７４、７６、７８、及び８０を停止することなくＭＸＵデータ・バス３９のビジー期間を考慮するためにストア・データ・バッファ５０２が設けられる。

16

【００３８】本発明によれば、ＭＸＵ５０は現在得られるベクトル・プロセッサに比べていくつかの明瞭な利点を与える。この利点を更に容易に理解するために、図２は現在得られるベクトル処理装置の概略図を示し、図３は本発明によるマトリクス処理装置の概略図を示す。図２及び図３の比較及び以下の説明は、それらの相違及び本発明の利点を指摘することになろう。

【００３９】図２は、計算機システム、即ち、メイン・メモリ及び中央処理装置２０２からデータ及び制御線２０４を介してデータ及び命令を受け取るベクトル処理装置２００の概略図を示す。ベクトル処理装置２００は、少なくとも１つの演算装置（ＡＵ）２０６を含んでいる。その演算装置２０６は、加算、乗算、及び論理的オペレーション等のようなオペレーションを遂行できる。ベクトル処理装置２００は複数個のベクトル・レジスタＲ１乃至ＲＮを有する。但し、Ｎは１よりも大きい。従って、そのシステムはＮ個のベクトル・レジスタを有する。これらレジスタのうちの４つが、図２において、レジスタ２０８、２１０、２１２、及び２１４として示される。各レジスタＲ１乃至ＲＮはワードＷ１乃至ＷＫを含んでいる。但し、Ｋは１よりも大きく、各ワードは１ビットよりも大きいデータを有する。従って、各レジスタはＫ個のワードを含んでいる。Ｎは、典型的には、８乃至３２個のレジスタ数である。データ又は命令はデータ線２０４、２１６、２１８、２２０及び２２２によって計算機システム２０２から転送される。各ワードは、各レジスタが長さＫのベクトルを形成するようにＫ個のワードのデータで満たされるまで、ワード・ロケーションＷ１乃至ＷＫに逐次に転送される。

【００４０】例えば、レジスタ２０８をレジスタ２１０に加算しそしてその値をレジスタ２１２に置くようなコマンドがＡＵ２０６によって実行される。各対応するレジスタ・ロケーションのワードが逐次に加算され、そしてその結果が図３における対応したワード・ロケーションに置かれる。例えば、レジスタＲ１のワード・ロケーションＷ１、即ち、ワード・ロケーションＷ１Ｒ１、におけるワードがデータ・バス２２４及び２２６を介してＡＵ２０６に転送される。レジスタＲ２のワード・ロケーションＷ１、即ち、ワード・ロケーションＷ１Ｒ２、におけるワードは、データ・バス２２８及び２３０を介してＡＵ２０６に転送される。Ｗ１Ｒ１＋Ｗ１Ｒ２のオペレーションがＡＵ２０６において遂行され、その結果は、データ線２３１及び２３２を介してレジスタＲ３のワード・ロケーションＷ１、即ち、ワード・ロケーションＷ１Ｒ３、に転送される。次のオペレーションは、Ｗ２Ｒ１におけるワードをＡＵ２０６に転送すること及びＷ２Ｒ２におけるワードをＡＵ２０６に転送することであり、ＡＵ２０６はＷ２Ｒ１＋Ｗ２Ｒ２のオペレーショ

17

ンを遂行してその結果をワード・ロケーションＷ２Ｒ３に入れる。ワード・ロケーションＷ１乃至ワード・ロケーションＷＫまで、各レジスタＲ１，Ｒ２におけるワード・ロケーションから逐次転送が行われる。従って、メイン・メモリ及び中央処理装置２０２は、２つ以上の転送オペレーション及び２つ以上の演算オペレーションが一時に行われるように、２つ以上の演算装置ＡＵを持つものであってもよい。

【００４１】典型的には、一般に使用されるマトリクス・ベクトル処理装置は、精々４つの演算装置ＡＵを持つものであり、それは、一時に４つのオペレーションを行うことができることを意味する。１つのベクトル・オペレーションは、レジスタＲ１におけるような１つのベクトルとレジスタＲ２におけるベクトルとの、加算、乗算又は論理的比較のような演算結合、及びＲ３のような他のベクトルへのその結果の転送である。そのようなベクトル命令又はオペレーションに対する機械サイクルの数は、そのような一般に得られるベクトル・プロセッサに対する５乃至１０機械サイクルとベクトル長Ｋを演算装置ＡＵの数により除したものとの和である、１つのベクトル命令に対する始動サイクルに等しい。その演算装置は、この分野では、演算パイプとも呼ばれる。そのようなベクトル・プロセッサにおけるベクトル・オペレーションのための最小のサイクル・タイムは、パイプの数がベクトル長に等しい時、即ち、Ｗ１乃至ＷＫのような各ワード・ロケーションがそれと関連付けられたパイプ又は演算装置を持つ時、である。そのような場合、１ベクトル命令当たりのサイクル・タイムは、１つのベクトル・オペレーションに対して、５乃至１０機械サイクル＋１である始動時間に等しい。１ベクトル命令当たりの最大のサイクル数は、パイプの数が１に等しい時に生じ、その場合、１ベクトル命令当たりのサイクル数は、５乃至１０機械サイクル＋ベクトル長Ｋである始動サイクルに等しい。

【００４２】図２に概略的に示される現在利用可能なベクトル処理装置は、計算機システム２０２及びベクトル処理装置２００の間のデータ帯域幅が不十分で、Ｋ個の演算装置（ＡＵ）２０６の能力を利用するに十分な程速くデータ又は命令を供給できないため、Ｋ個の演算装置２０６を持つことができない。

【００４３】拡張帯域幅のメモリは、ベクトル・オペレーション相互間で５乃至１０サイクルの最少時間を得るためには、Ｋ個の演算装置２０６を利用する必要があろう。Ｑが１つの演算装置２０６に対する帯域幅である場合、Ｋ個の演算装置２０６に対する帯域幅はＫ＊Ｑでなければならない。そのような拡張帯域幅のメモリは、大きく増加した複雑性と、それによるコストとを必要とする。反対に、本発明によるプロセッサは、拡張帯域幅を必要とせずにベクトル・オペレーション相互間で１サイクル・タイムを得るものであり、Ｑの帯域幅でもってこ

18

れを得ることができる。

【００４４】１つのベクトルの負の逆元又は絶対値、２つのベクトルの加算又は乗算、又はこれらの演算オペレーションの組合せのようなベクトル・オペレーションが、少なくとも１つ（代表的には、２つ又は３つ）のベクトルにおいて遂行される。ベクトル・オペレーションは、２つのベクトルを乗算しそして第３のベクトルを加算するオペレーション（１つのベクトル・オペレーションで行われる）のような３つ以上のベクトルに関するものであってもよい。

【００４５】本発明によるベクトル処理装置、又はマトリクス処理装置３００が図３に３００として概略的に示される。処理装置３００は、計算機システム３０２からデータ及び制御線３０４を介してデータ及び命令を受け取る。Ｎ個のベクトルに対応したＮ個のレジスタＲ１乃至ＲＮがあり、Ｋのベクトル長に対してこれらレジスタの各々におけるＫ個のワードＷ１乃至ＷＫがある。レジスタＲ１乃至ＲＮの各々における第１ワード・ロケーションＷ１はすべて、３０３として示された処理エレメント１（ＰＥ１）に含まれている。処理エレメント３０３は演算装置１（ＡＵ１）３０５を含む。同様に、レジスタＲ１乃至ＲＮの各々における第２ワードＷ２は、３０６として指定された処理エレメントＰＥ２に含まれる。処理エレメント３０６は、３０８として示された演算装置ＡＵ２を含む。同様に、それらベクトルの各々における各ワードエレメントＷＩ（但し、ＩはＫまでである）は、それ自身の演算装置を含む処理エレメント内に含まれる。各ワード・エレメントＷＩは、Ｍビット・ワードに対するＭビットのデータを記憶するためのＭ個のロケーションを有するレジスタに含まれる。各々がＫ個のワードを有するＮ個のベクトルは、Ｋ＊Ｎワード・マトリクスを形成する。従って、ここでは、図３のプロセッサはマトリクス・プロセッサと呼ばれる。

【００４６】図１に戻ると、マトリクス装置５０は、マトリクス制御装置ＭＸＣＵ５１及びＫ個の処理エレメントＭＸＰＥ１乃至ＭＸＰＥＫを含む。これらマトリクス処理エレメントのうちの６つは数字６０、６１、６２、６３、６４、及び６５によって識別される。マトリクス制御装置ＭＸＣＵ５１のエレメント５２は、コマンド・バス７０及び複数個のデータ制御バス６８を介してマトリクス・エレメントＭＸＰＥ１乃至ＭＸＰＥＫの各々に接続される。コマンド・バス７０は、マトリクス制御装置５２から処理エレメントＭＸＰＥ１乃至ＭＸＰＥＫの各々にコマンドを搬送する。図１では、データ制御バス６８は、エレメント７４、７６、７８、及び８０として示される。図１に示されるように、データ制御バス６８のうちのいくつかは、２つ以上のマトリクス処理エレメントＭＸＰＥの間で共用可能である。例えば、データ制御バス８０は、データ制御線８１及び８２によって示されるように、それぞれ、マトリクス処理エレメントＭＸ

19

PE1及びMXPE5の間で共用される。ベクトル・レジスタ402は、図3における単一の処理エレメント303のレジスタ、例えば、ワード・ロケーションW1R1乃至W1RNに対応する。

【0047】図5は、図1のマトリクス制御装置（MXCU）51の概略図を示す。命令は、MXU50から命令バス41を介して制御装置570により受け取られる。それは、命令のタイプに基づいて、データをストア・データ・バッファ502に記憶すべきか、又はロード・データ・バッファ504に記憶すべきかというそれを扱う方法を決定する。データは、CPUデータ・バス39を介してMXU50に及びMXU50から転送される。CPUデータ・バス39を介して到着するデータはマルチプレクサ501に入る。そのマルチプレクサは、CPU45から情報を受け取る時、バス506を介してロード・データ・バッファ504にそのデータを経路指定する。ロード・データ・バッファ504におけるデータは、スイッチ、即ち、ルータ508によって、図1のデータ制御バス68に対応したデータ・バス510、512、514、及び516に分けられる。各データ・バス510、512、514、及び516は、それぞれ、マルチプレクサ518、520、522、及び524に通じ、そしてそれらマルチプレクサから、データがデータ・バス74、76、78、及び80に転送される。データ・バス74、76、78、及び80におけるそのデータは、マトリクス処理エレメントMXPE1乃至MXPEKに転送される。

【0048】一方、データがマトリクス処理エレメントMXPE1乃至MXPEKから転送される時、そのデータはデータ・バス74、76、78、及び80を介して、それぞれ、マルチプレクサ518、520、522、及び524に転送され、そしてそこから、それぞれ、データ線526、528、530、及び532を介してスイッチ、即ち、ルータ534に送られる。データは、そのスイッチ534からストア・データ・バッファ502に転送され、更に、データ線535、マルチプレクサ501、及びデータ・バス39を介して図1のCPU45に転送される。マルチプレクサ501は、1つの双方向入力、即ち、データ・バス39と、2つの単方向データ線535及び506を有する。

【0049】図4は、図1のMXU50におけるMXPEの1つの概略図を更に詳細に示す。データ・バス401は、図1のデータ・バス74、76、78、又は80のうちの任意のものでよい。NICAがそのシステムの残りのもの（メモリ）と対話する方法を以下で説明する。本発明によるマトリクス装置の1つの主要な目的は、メモリ帯域幅（BW）要件を減少させることである。これは、NICAレジスタにロードされたデータの有意味な再使用によって達成可能である。従って、NICAとメモリとの間の帯域幅（BW）に関する設計ポイントは、1サイクル当たりのベクトル・ロード／ストア（Kオペレーション）よりもかなり少なくなければならない。ベクトル・ロード／ストアは、多数のサイクルにおいて1回の割合でしか起こらないものと予測される。

20

【0050】この仮定の場合、NICAとシステムとの間のデータ転送を2つの要素に分割することができる。第1の要素は、システムとNICAロード・ストア／バッファ（LSB）400との間の非同期転送である。第2の要素は、LSB400とNICAベクトル・レジスタ402との間の同期（SIMD）転送である。ちょうどベクトル・レジスタのように、LSB400も、MXU50におけるすべてのMXPEに分布している。LSB400における1つのMXPE当たりの与えられたロケーションの数がもう1つの設計パラメータである。ベクトル・ロードに対して、システムは、メモリにおけるオペランドのアドレスを計算し、それらをフェッチし、そして宛先MXPEのインデックスをもって到着データをタグし、それらをバス上に（高い帯域幅に対しては、各バスがMXPEのサブセットにだけ接続されて成る複数のバスが使用可能である）置く。宛先MXPEは到着データをラッチし、それをLSB400の指定されたロケーションに置く。すべてのMXPEがそれらのデータを受け取った時、MXPE制御装置404は、LSB400からその指定されたベクトル・レジスタへのデータのSIMD転送を開始する。ベクトル・ストアは同様に、しかし、逆に作用する。データ再使用のために、ベクトル・ロード／ストアは滅多にないことであり、NICAはレジスタにおけるデータに関してレジスタ・ツー・レジスタ（RR）ベクトル演算を行い続けるものと考えられる。

【0051】次のものは拡張の例示リストである。これらの思想のいくつか又はすべてがNICAにおいて利用可能である。即ち、

○「スカラ・レジスタ」のセットは、別個のスカラ・レジスタ・バスを介してすべてのMXPEに対して利用可能にされる。バスはスカラ・レジスタのプールから送られ、すべてのMXPEは各MXPE上の別個の「スカラ・ポート」を介してそれに接続される。このバスは、すべてのPEの演算装置にスカラ定数を与えるために使用可能である。このバスは、ベクトル・レジスタにおける定数の速いローディングのためにも使用可能である。

○多数のPEが単一のチップ／モジュール上に製作可能である。これは、すべてのバスに対するI／OピンがPE間で共用可能であるという利点を有する。

○M個の1ビット・ベクトル・マスク又は条件レジスタのセットを設けることが可能である。これらは、再び、各PEがM個の1ビット・マスク又は条件レジスタのセットを有する場合、すべてのPEに分配される。これらのレジスタは、比較によって、又は演算オペレーションの条件コードによってセット可能である。それらは、P

21

Ｅとそのシステムとの間のデータ移動を実施するために並びにＰＥ内での条件付き演算及びデータ移動を実施するために使用可能である。これらのビット自体は、システムとＮＩＣＡとの間で双方向に転送可能である。ベクトル条件レジスタとシステムとの間のビット転送のために、別個のＫビット幅の並列パスが使用可能である。各ＰＥは個の並列パスのうちの１ビットを得る。これらのマスクがＮＩＣＡとシステムとの間のデータ転送を制御するために使用される場合、制御装置にとってこのパスが利用可能であることは、その転送のための制御データの速い先回り処理を助けるであろう。マスク・ベクトルは1ビットのブール演算によっても操作可能である。

○パック／アンパック・オペレーションを行うと（例えば、２つの短精度オペランドを１つの長精度オペランドとしてパックする）、短精度データはバス上を更に効率的に転送可能である。これは、利用可能なメモリ及びバス帯域幅の効率的な利用を助ける。

○ＳＩＭＤベクトル命令に参加するＭＸＰＥの範囲は、アクティブナＭＸＰＥの開始及び終了インデックスを指定する「ベクトル範囲設定」命令によって制御可能である。この範囲内であっても、対応するＭＸＰＥにおける指定されたマスク・ビット（M個のマスク・ビットの１つ）が「1」である場合しか目標は更新されない。

【００５２】ＮＩＣＡがＮＩＣ問題を解決する場合に適用可能であるためには、２つの主要な必要条件、即ち、ベクトル化及びデータ再使用、がある。最も伝統的なベクトル・オペレーションは、ＮＩＣＡにおいてＳＩＭＤ態様で実施可能である。しかし、いくつかの例外がある。1つのベクトルにおける相異なるエレメントの間の相互作用を必要とするベクトル・オペレーションをＳＩＭＤ態様で行うことはできない。そのような命令の重要な例はベクトル累算（乗算累算は累算と同じである）である。累算では、ベクトルのサム・リダクション（ｓｕｍ　ｒｅｄｕｃｔｉｏｎ）が行われる。演算パイプラインのため、ベクトル・マシン上でも、これはあまり効率的ではない。非常に長いベクトルを累算する場合、そのベクトルの８つ毎のエレメントがアキュムレータの１つにおいて累算される。オペレーションのこのフェーズは十分なベクトル速度で完了する。これに続いて、「部分和の合計」オペレーションが生じる。それはスカラ・モードで行われる。累算（又は、乗算累算）は、部分和の数がＫに等しくされた場合、ＮＩＣＡ上でも実施可能である。次に、非常に長いベクトルに対しては、殆どの計算が高パフォーマンスのＳＩＭＤ態様で行われるであろう。最後の「部分和の合計」位相だけがスカラ計算で行われるであろう。乗算累算オペレーションを回避するために、しばしば、殆どのアルゴリズムが再公式化可能であり、そしてその代わりに、乗算加算オペレーションが使用可能である。

【００５３】ＳＩＭＤ計算を施すことが可能でないもう

22

１つのオペレーションは、ベクトルの最小及び最大を見つけることである。ここで、再び、問題サイズをＫまで縮小するためにＳＩＭＤベクトル比較が使用可能である。従って、その作業の殆どが高パフォーマンスＳＩＭＤモードで遂行可能である。

【００５４】前述のように、ベクトル化はＮＩＣＡを使用する場合の前提である。殆どのＮＩＣＡアプリケーションが内部ループ・レベル又は外部ループ・レベルでベクトル化する。ＮＩＣＡが非常に効果的であるためには、ベクトル・レジスタに与えられたデータの有意味な再使用が必要である。再使用は、浮動小数点演算対データ比（ＦＤＲ）に関連する。当初の問題を或レベルでブロック化することによって、多くの問題に対して、ＦＤＲは大いに改良可能である。１つの有用な技法は外部ループ・ベクトル化である。その場合、内部ループ又はこれらのサブブロックがＮＩＣＡレジスタに適合する。そこで、外部ループがＫのブロック・サイズでもってブロック化され、そしてすべての内部ループ計算が１つのＰＥにおいて生じる時、外部ループの各インデックスが別個のＰＥにおいて実行される。これは、外部ループ・ベクトル化／並列化の非常に一般的な形式である。多くの並列化コンパイラが、この技法を使用して多くの並列プロセッサに跨って計算を分配する。ベクトル化し得ない多くの問題が、しばしば、並列化可能である。例えば、地震の計算では、複雑な三重対角等式の多重システムを解く場合に、大量の計算が行われる。これらは、パフォーマンスにおける非常に大きな利得をもって、ＮＩＣＡにおいて実施可能である。大量の地震の計算はトレース・ベース化され、しばしば、それらトレースに跨って並列化可能である。

【００５５】ＦＯＲＴＲＡＮ－９０と呼ばれるＦＯＲＴＲＡＮの拡張が定義されている。それは、アレイ演算を行う機能を提供する。ＦＯＲＴＲＡＮ－９０で書かれたプログラムはＮＩＣＡにとって容易にコンパイル可能である。

【００５６】ＬＡＰＡＣＫは、線形代数計算を行うための公知のソフトウエアである。それは、ＬＩＮＰＡＣＫ及びＥＩＳＰＡＣＫパッケージに取って代わるものであり、それらを大いに拡張するものである。それは、殆どの現在利用可能なコンピュータにおける非常に高いパフォーマンスのために書かれている。それの高いパフォーマンスのために、線形代数サブルーチン及びＮＡＧのような他の商業計算ソフトウエア・パッケージの殆どのユーザが、ＬＡＰＡＣＫに移行している。ＬＡＰＡＣＫプロジェクトの背後にある重要な思想は、殆どの計算がＢＬＡＳ－３で行われない場合、高パフォーマンスを得ることはできないということである。ＢＬＡＳ－３は、$N^2$ のオーダのデータ・ポイントで、$N^3$ のオーダの計算を行う線形代数カーネルのセットである。ＬＡＰＡＣＫコードは移植可能であり、コンピュータ製造者によって

提供された調整された（特殊機械用に）ＢＬＡＳ－３カーネルを利用する。この思想及びソフトウエア・パッケージは、共用メモリ並列プロセッサ及びＭＩＭＤ機械まで更に拡張される。ＢＬＡＳ－３ルーチンの調整されたセットがＮＩＣＡ用に開発されることが期待される。そこで、ＬＡＰＡＣＫサブルーチン・コールを持ったユーザ・コードは、ユーザ部分において如何なる努力も払うことなく、ＮＩＣＡからパフォーマンス・ブーストを自動的に得るであろう。

【００５７】一方、ＢＬＡＣ－３ルーチンは、ＤＧＥＭＭ（更新を伴うマトリクス・マトリクス乗算）においてその計算の殆どを行う。ＤＧＥＭＭにおいて行われない計算は、それが三角マトリクスにおいて行われることを除けば同じである。以下では、ＮＩＣＡにおいて非常に効率的にＤＧＥＭＭを実施できる方法を示す。

【００５８】ＤＧＥＭＭは次の計算、即ち、

Ｃ<－Ｃ＋Ａ＊Ｂ

を行う。

【００５９】サイズがＫ＊ＰであるＣというブロックが一時に計算されるように、Ｃに関する何らかの外部レベル・ブロッキングがあるものと仮定する。但し、ＰはＮよりも小さく、ＮＩＣＡにおいて利用可能なベクトル・レジスタの数である。その場合、Ｃマトリクスのこのブロックはレジスタに適合する。Ａはサイズ　Ｋ＊Ｌのものであると仮定する。但し、Ｌはその問題の中間の大きさである。そこで、上記の計算は、Ｃマトリクスに関するランクＬ更新を行うことになる。これは、Ｌ個のステップで行うことができる。各ランクＩ更新のためには、Ａマトリクス（Ｋ個のエレメント）の１つの列とＢマトリクス（Ｐ個のエレメント）の１つの行が必要である。これらＫ＋Ｐ個のエレメントがＮＩＣＡに与えられ、２＊Ｐ＊Ｋ浮動小数点演算を生成するために使用される。この問題に対する浮動小数点演算対データの比は、２/（１/Ｋ＋１/Ｐ）である。Ｋ及びＰの両方とも大きい場合、これは非常に計算集中的な問題となる。ＮＩＣＡとシステムとの間のメモリ帯域幅は、２＊Ｋ＊Ｐ浮動小数点演算を行いながら（Ｋ＋Ｐ）データ転送をサポートするに十分な大きさでなければならない。２＊Ｋ浮動小数点演算が１サイクルで行い得るものであると仮定すると、１サイクル当たり少なくとも（１＋Ｋ/Ｐ）個のエレメントのデータ転送速度を必要とする。これは、Ｃマトリクスの初期ローディング及び最終ストアリングを考慮していない。Ｌが大きい場合、この余分なオーバヘッドは小さくなる。

【００６０】１次元及び多次元ＦＥＴ、多重シーケンスの回旋／相関、地震の処理における３次元移行等のような多くの他の重要な計算カーネルが知られている。これらアプリケーションはすべて、ＮＩＣＡから大きなパフォーマンス利得を得ることができる。一般的な稠密、正定値の対称、複体対称、帯域（ＶＳＳのオーダの、又は



更に長い帯域幅を持つ）、スカイライン等のような種々な種類の式を解くことは、すべて、ピークに近いパフォーマンスでもってＮＩＣＡ上で実施可能である。汎用稠密システムの等式（サイズ１０００又はそれ以上）を解くＬＩＮＰＡＣＫ　ＴＰＰベンチマークも、ピーク・パフォーマンスに近くなるであろう。これは非常に重要な超高速計算ベンチマークであり、スーパコンピュータは、しばしば、この基準で比較される。

【００６１】大規模な科学計算におけるもう１つの傾向は、非常に大きな疎システムの等式に向いている。これらの問題を解くために、しばしば、直接方法が使用される。直接疎方法は疎ＢＬＡＳ－３（現在の標準的活動はこれらＢＬＡＳを定義する用に進んでいる）によって公式化可能である。一方、疎ＢＬＡＳ－３は、ＮＩＣＡに対してブロック化可能である。疎マトリクスは、抽出されたブロックが全く稠密に見えるような方法で行及び列のセットがそのマトリクスから抽出されるように再配列可能である。疎マトリクス全体は、これらブロック及び他の構造体（対角線の帯域のような）のセットの線形合計として表される。これらは稠密ブロックとしてＮＩＣＡベクトル・レジスタに転送され、何回も使用可能である。

【００６２】高パフォーマンスのグラフィックスも高いＦＤＲを持つ傾向があり、ベクトル化可能である。これらのアプリケーションもＮＩＣＡから利益を得ることができる。

【００６３】まとめとして、本発明の構成に関して以下の事項を開示する。

【００６４】（１）プロセッサ・サイクル・タイムを有する計算機プロセッサにして、Ｋ個の処理エレメント（但し、Ｋ＞１）にして、前記Ｋ個の処理エレメントの各々はＮ個のレジスタのセット（但し、Ｎ＞１）及び演算装置を含み、前記レジスタの各々のＩ番目のエレメントはベクトル長ＫのＩ番目のベクトルのための記憶ロケーションを提供し（但し、１≦Ｉ≦Ｎ）、それによって、Ｎ個のベクトルを提供するものと、前記Ｎ個のベクトルの少なくとも１つの間でベクトル・オペレーションを所定のサイクル・タイムで遂行し、前記ベクトル・オペレーションの結果を前記Ｎ個のベクトルの１つに記憶するための手段にして、前記ベクトル・オペレーションが前記Ｎ個のベクトルの少なくとも１つにおけるＫ個のエレメントすべてに対して実質的に同時に前記Ｋ個の演算装置によって遂行されるものと、前記所定のサイクル・タイム後の次のサイクルタイムにおいて他のベクトル・オペレーションを遂行するための手段と、を含む計算機プロセッサ。

（２）前記Ｎ個のレジスタのセットにおける各レジスタはＭ個のビット記憶ロケーション（但し、Ｍ＞０）を有することを特徴とする上記（１）に記載の計算機プロセッサ。

25

（3）前記ベクトル・オペレーションを遂行するための手段である第1演算装置及び前記他のベクトル・オペレーションを遂行するための手段である第2演算装置は同じ演算装置であることを特徴とする上記（1）に記載の計算機プロセッサ。

（4）マトリクス制御装置を含むことを特徴とする上記（1）に記載の計算機プロセッサ。

（5）前記マトリクス制御装置を前記K個の処理エレメントに接続するコマンド・バスを含むことを特徴とする上記（4）に記載の計算機プロセッサ。

（6）前記マトリクス制御装置を前記K個の処理エレメントに接続する少なくとも1つのデータ・バスを含むことを特徴とする上記（4）又は（5）に記載の計算機プロセッサ。

（7）前記少なくとも1つのデータ・バスは前記K個の処理エレメントのうちの2個以上の処理エレメントによって共用されることを特徴とする上記（5）に記載の計算機プロセッサ。

（8）サイクル・タイムを有する計算機プロセッサにして、マトリクス制御装置と、前記サイクル・タイムのうちの先行のベクトル・オペレーションの1サイクルにおいて後続のベクトル・オペレーションを遂行するための手段と、を含む計算機プロセッサ。

（9）前記手段はベクトル演算オペレーションを遂行することを特徴とする上記（8）に記載の計算機プロセッサ。

（10）前記マトリクス制御装置を前記K個の処理エレメントに接続するコマンド・バスと、前記マトリクス制御装置を前記K個の処理エレメントに接続する少なくとも1つのデータ・バスと、を含むことを特徴とする上記（8）に記載の計算機プロセッサ。

（11）サイクル・タイムを有する計算機システムにして、メイン・メモリと、キャッシュ・メモリと、記憶制御装置と、中央処理装置と、マトリクス処理装置と、前記中央処理装置から前記マトリクス処理装置に命令を転送するためのマトリクス命令パスと、前記キャッシュ・メモリから前記マトリクス処理装置にデータを転送するための中央処理装置データ・パスと、前記中央処理装置は前記記憶制御装置と前記キャッシュ・メモリとを介して前記メイン・メモリに接続する手段と、前記マトリクス制御装置は前記キャッシュ・メモリを介して前記メイン・メモリに接続する手段と、前記記憶制御装置はメモリ・アドレス・バスによって前記メイン・メモリに接続する手段と、前記記憶制御装置は第1データ・バスによって前記中央処理装置に接続する手段と、前記キャッシュ・メモリは第2データ・バスによって前記マトリクス処理装置に接続する手段と、前記マトリクス処理装置は前記サイクル・タイムのうちの先行のベクトル・オペレーションの1サイクルにおいて後続のベクトル・オペレーションを遂行するための手段を含むことと、K個の処

26

理エレメント（但し、K＞1）にして、前記K個の処理エレメントの各々はN個のレジスタのセット（但し、N＞1）及び演算装置を含み、前記レジスタの各々のI番目のエレメントはベクトル長KのI番目のベクトルのための記憶ロケーションを提供し（但し、1≦I≦N）、それによって、N個のベクトルを提供するものと、前記N個のベクトルの少なくとも1つの間でベクトル・オペレーションを所定のサイクル・タイムで遂行し、前記ベクトル・オペレーションの結果を前記N個のベクトルの1つに記憶するための手段にして、前記ベクトル・オペレーションが前記N個のベクトルの少なくとも1つにおけるK個のエレメントすべてに対して実質的に同時に前記K個の演算装置によって遂行されるものと、前記所定のサイクル・タイム後の次のサイクルタイムにおいて他のベクトル・オペレーションを遂行するための手段と、を含む計算機システム。

（12）サイクル・タイムを有する計算機システムにして、メイン・メモリと、キャッシュ・メモリと、記憶制御装置と、中央処理装置と、マトリクス処理装置と、前記中央処理装置から前記マトリクス処理装置に命令を転送するためのマトリクス命令パスと、前記キャッシュ・メモリから前記マトリクス処理装置にデータを転送するための中央処理装置データ・パスと、を含み、前記中央処理装置は前記記憶制御装置と前記キャッシュ・メモリとを介して前記メイン・メモリに接続する手段と、前記マトリクス制御装置は前記キャッシュ・メモリを介して前記メイン・メモリに接続する手段と、前記記憶制御装置はメモリ・アドレス・バスによって前記メイン・メモリに接続する手段と、前記記憶制御装置は第1データ・バスによって前記中央処理装置に接続する手段と、前記キャッシュ・メモリは第2データ・バスによって前記マトリクス処理装置に接続する手段と、前記マトリクス処理装置は、K個の処理エレメント（但し、K＞1）にして、前記K個の処理エレメントの各々はN個のレジスタのセット（但し、N＞1）及び演算装置を含み、前記レジスタの各々のI番目のエレメントはベクトル長KのI番目のベクトルのための記憶ロケーションを提供し（但し、1≦I≦N）、それによって、N個のベクトルを提供するものと、前記N個のベクトルの少なくとも1つの間でベクトル・オペレーションを所定のサイクル・タイムで遂行し、前記ベクトル・オペレーションの結果を前記N個のベクトルの1つに記憶するための手段にして、前記ベクトル・オペレーションが前記N個のベクトルの少なくとも1つにおけるK個のエレメントすべてに対して実質的に同時に前記K個の演算装置によって遂行されるものと、前記所定のサイクル・タイム後の次のサイクルタイムにおいて他のベクトル・オペレーションを遂行するための手段と、を含むことを、特徴とする計算機システム。

（13）マトリクス制御装置と、前記マトリクス制御装

27

置を前記Ｋ個の処理エレメントに接続するためのコマンド・バスと、前記マトリクス制御装置を前記Ｋ個の処理エレメントに接続するための少なくとも１つのデータ・バスと、を含むことを特徴とする上記（１２）に記載の計算機システム。

（１４）サイクル・タイムを有する計算機システムにして、メイン・メモリと、中央処理装置と、マトリクス処理装置と、前記中央処理装置から前記マトリクス処理装置に命令を転送するためのマトリクス命令バスと、前記メイン・メモリから前記マトリクス処理装置にデータを転送するための中央処理装置データ・バスと、前記中央処理装置は第１データ・バスによって前記メイン・メモリに接続する手段と、前記マトリクス制御装置は第２データ・バスによって前記メイン・メモリに接続する手段と、前記記憶制御装置はメモリ・アドレス・バスによって前記メイン・メモリに接続する手段と、前記マトリクス処理装置は前記サイクル・タイムのうちの先行のベクトル・オペレーションの１サイクルにおいて後続のベクトル・オペレーションを遂行するための手段を含むことと、Ｋ個の処理エレメント（但し、Ｋ＞１）にして、前記Ｋ個の処理エレメントの各々はＮ個のレジスタのセット（但し、Ｎ＞１）及び演算装置を含み、前記レジスタの各々のＩ番目のエレメントはベクトル長ＫのＩ番目のベクトルのための記憶ロケーションを提供し（但し、１≦Ｉ≦Ｎ）、それによって、Ｎ個のベクトルを提供するものと、前記Ｎ個のベクトルの少なくとも１つの間でベクトル・オペレーションを所定のサイクル・タイムで遂行し、前記ベクトル・オペレーションの結果を前記Ｎ個のベクトルの１つに記憶するための手段にして、前記ベクトル・オペレーションが前記Ｎ個のベクトルの少なくとも１つにおけるＫ個のエレメントすべてに対して実質的に同時に前記Ｋ個の演算装置によって遂行されるものと、前記所定のサイクル・タイム後の次のサイクルタイムにおいて他のベクトル・オペレーションを遂行するための手段と、を含む計算機システム。

28

【００６５】

【発明の効果】本発明により、計算機システムのメモリ帯域幅を大きく増大させることなく、制限された数の命令をより速く実行することが可能となる。

【図面の簡単な説明】

【図１】本発明の実施例に従って構成されたデータ処理システムのブロック図を示す。

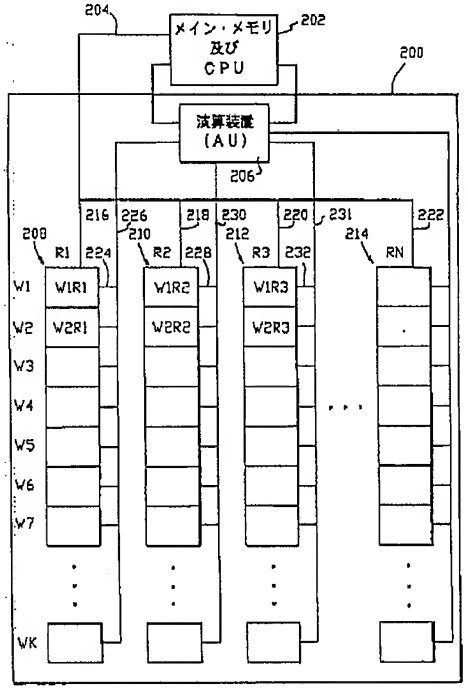
【図２】一般に利用可能なベクトル・プロセッサの概略図を示す。

【図３】本発明によるベクトル・プロセッサの概略図である。

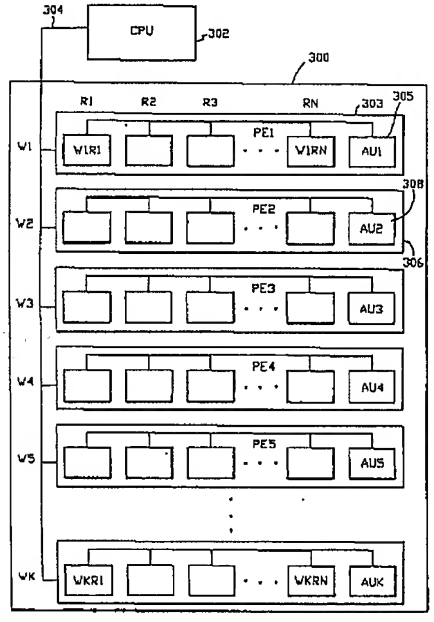
【図４】図３のベクトル・プロセッサの処理エレメントＭＸＰＥの１つの詳細なブロック図である。

【図５】図３のベクトル・プロセッサのマトリクス制御装置ＭＸＵの詳細なブロック図である。

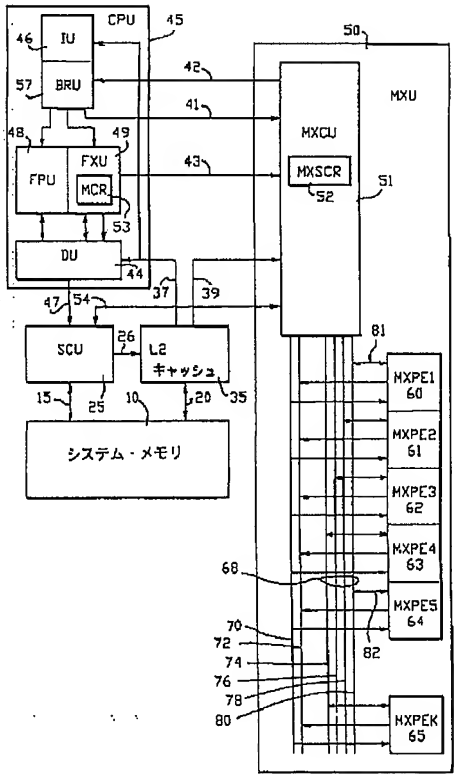
【図２】



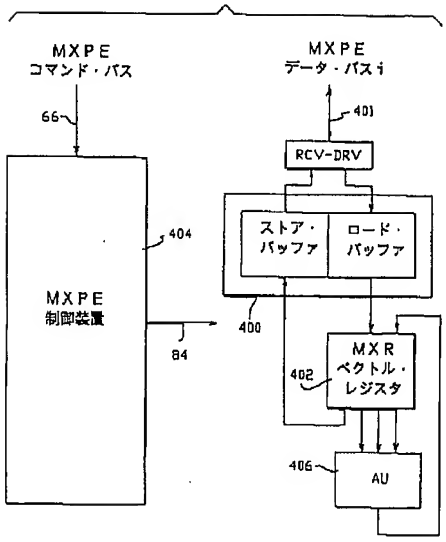

【図３】



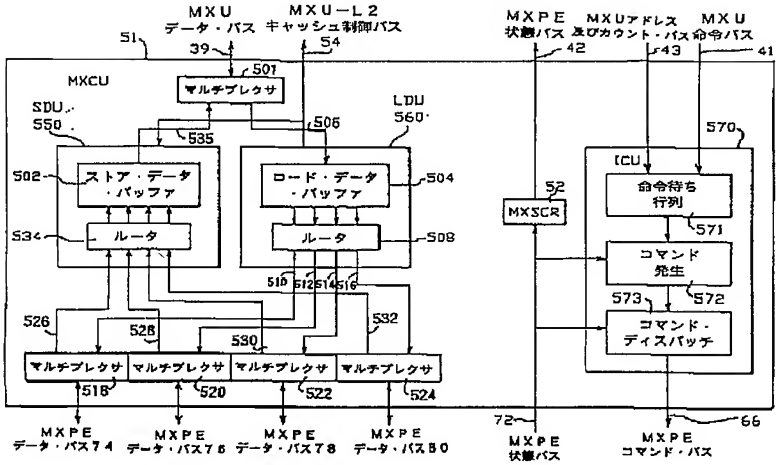
【図1】


CPU
45
IU
46
BRU
57
48
49
FPU
FXU
MCR
53
DU
44
47
54
SCU
26
L2
キャッシュ
15
25
10
20
35
システム・メモリ
50
MXU
42
41
MXCU
43
MXSCR
52
51
37
39
81
MXPE1
60
MXPE2
61
MXPE3
62
MXPE4
63
MXPE5
64
MXPEK
65
68
70
72
74
76
78
80
82

【図4】


MXPE
コマンド・バス
66
MXPE
データ・バス
401
RCV-DRV
ストア・
バッファ
ロード・
バッファ
404
MXPE
制御装置
400
84
MXR
ベクトル・
レジスタ
402
406
AU

【図5】


MXU
データ・バス
MXU－L2
キャッシュ制御バス
MXPE
状態バス
MXUアドレス
及びカウント・バス
MXU
命令バス
51
39
54
42
43
41
MXCU
501
マルチプレクサ
SDU
550
LDU
560
506
535
570
ICU
502
ストア・データ
・バッファ
ロード・データ
・バッファ
504
52
MXSCR
命令待ち
行列
571
534
ルータ
ルータ
508
コマンド
発生
510
512
514
516
573
572
526
528
530
532
コマンド・
ディスパッチ
マルチプレクサ
マルチプレクサ
マルチプレクサ
マルチプレクサ
518
520
522
524
72
66
MXPE
データ・バス74
MXPE
データ・バス76
MXPE
データ・バス78
MXPE
データ・バス80
MXPE
状態バス
MXPE
コマンド・バス

フロントページの続き

(72)発明者　ランダル・ディーン・グローブス
アメリカ合衆国テキサス州、オースティン、ミドルビィー・ドライブ　9002
(72)発明者　フレッド・ゲールング・ガスタブソン
アメリカ合衆国ニューヨーク州、ブリアクリフ・マノア、サウス・スティト・ロード　70
(72)発明者　マーク・アラン・ジョンソン
アメリカ合衆国テキサス州、オースティン、タレイラン・ドライブ　10105
(72)発明者　ブレット・オルッソン
アメリカ合衆国テキサス州、ラウンド・ロック、シンコタニュー・ウェイ　1800